BEAR: Physics-Principled Building Environment for Control and Reinforcement Learning

Chi Zhang, Yuanyuan Shi
Department of Electrical and Computer Engineering
University of California San Diego

Yize Chen
AI Thrust, Information Hub
Hong Kong University of Science and Technology

ABSTRACT

Recent advancements in reinforcement learning algorithms have opened doors for researchers to operate and optimize building energy management systems autonomously. However, the lack of an easily configurable building dynamical model and energy management task simulation and evaluation platform has arguably slowed the progress in developing advanced and dedicated reinforcement learning (RL) and control algorithms for building operation tasks. Here we propose "BEAR", a physics-principled Building Environment for Control and Reinforcement Learning. The platform allows researchers to benchmark both model-based and model-free controllers using a broad collection of standard building models in Python without co-simulation using external building simulators. In this paper, we discuss the design of this platform and compare it with other existing building simulation frameworks. We demonstrate the compatibility and performance of BEAR with different controllers, including both model predictive control (MPC) and several state-of-the-art RL methods with two case studies. BEAR is available at https://github.com/chz056/BEAR.

CCS CONCEPTS

• **Hardware** → **Smart grid**; *Temperature simulation and estimation*; • **Theory of computation** → **Reinforcement learning**.

KEYWORDS

Building energy management; reinforcement learning;

1 INTRODUCTION

Building is one of the major sources of global energy consumption. In 2021, residential and commercial buildings were responsible for around 39% of total U.S. energy consumption and 74% of total U.S. electricity consumption [11]. Consequently, research on the operation of building Heating Ventilation and Air Conditioning (HVAC) systems can lead to significant energy savings and carbon emission reduction. Many control methods have been developed to provide solutions for building HVAC control problems, including model predictive control, nonlinear adaptive control, and decentralized control [7][17]. However, most such approaches require detailed and exact building dynamics models, and an increase in the complexity of building dynamics would lead to significantly higher computational costs. As a result, reinforcement learning (RL) has

gained tremendous interest for building control in modern days due to its model-free nature (see [8] for a recent review).

One challenge of the building RL research is the lack of a benchmarking simulation environment for developing and evaluating different RL algorithms with realistic building models. Several recent works [1, 6, 15, 18, 20] have proposed simulation solutions to address such a problem. However, most of them adopted a co-simulation framework with a python interface for algorithm development and an outsourcing building simulator, like Energy-Plus [3] or Modelica [9]. For researchers who do not yet have detailed knowledge of such packages, it is hard to test with their own configurations and validate RL performance. BOPTESTS-Gym [1], Sinergym [6], and Gym-Eplus [20] rely on EnergyPlus or Modelica to perform simulation, and the BCVTB middleware [19] to communicate between simulators and the platform interface. Energym [15] uses predefined building models and co-simulation with EnergyPlus. CityLearn [18] is almost self-contained that it uses pre-simulated data. However, CityLearn focuses on building-level control interacting with the grid, rather than zone-level detailed building simulation.

In this paper, we present BEAR, a physics-principled Building Environ-ment for control And Reinforcement learning. BEAR constructs building simulation from first-principled physics models and provides a scalable platform for researchers from different backgrounds to design, test and evaluate their reinforcement learning (RL) and control algorithms. BEAR can set up customized building environments by either choosing from a curated list of (building type, weather type, and city), or incorporating building and weather datasets of their own. To address the potential gap between the simulated configurations and the actual building dynamics, BEAR users can also efficiently train a data-driven model using self-collected dataset of their own. The proposed simulator supports fine-grained dynamics simulation and provides an OpenAI Gym interface [2] for developing RL agorithms. Researchers from the machine learning and RL community can design new environments and algorithms with minimal knowledge of the underlying building physical models and thus can focus more on algorithm development and evaluation. On the other hand, BEAR, with a physics-principled simulation engine, provides researchers and engineers an accessible platform to implement new building models with user-defined building structure, operation schedule, temperature, and other environmental variables. The primary characteristic that distinguishes BEAR from other building RL simulators is the physics-based modeling procedures as described in Section 2. Table 1 compares BEAR with some other RL environments for building control.

	BEAR	Sinergym [6]	Energym [15]	Gym-Eplus [20]	Citylearn [18]	RL Testbed for Energy-Plus [10]	BOPTESTS-Gym [1]
Simulator	Self-designed	EnergyPlus	EnergyPlus	EnergyPlus	Data	EnergyPlus	Modelica
Available Buildings	16+	3	7	1	9	1	6
Weather Types	19	3	4	1	5	5	5
Action Space	Both	Both	Discrete	Discrete	Continuous	Continuous	Both
Action Type	Energy	Temperature	Temperature	Temperature	Energy	Temperature & Fan flow rate	Temperature & Lower level actuator signals
Reward	Customized	Customized	Customized	Customized	Predefined	Predefined	Customized
Multi-agent	User-defined	No	No	No	Yes	No	No
Control Objectives	Energy demand, Thermal comfort	Energy demand, Thermal comfort	Grid exchange, Energy demand, CO2 emissions	Energy demand, Thermal comfort	Energy demand	Energy demand, Thermal comfort	Energy demand, Thermal comfort
Control Step	User-defined	User-defined	User-defined	5 minutes	1 hour	15 minutes	User-defined
Zone level Control	Yes	No	Yes	No	No	Yes	Yes

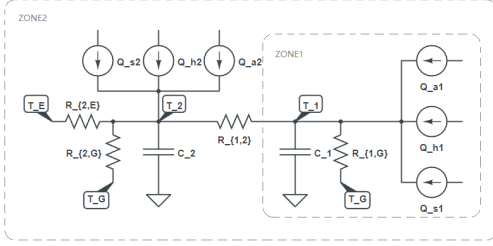**Table 1: Features of building control simulators.**

Figure 1: RC Model for a two-zone, one-story building.

2 BUILDING DYNAMICS

The Reduced Resistance-Capacitance (RC) model [7] is widely used for the building HVAC system model to simplify design complexity and reduce computation time. We construct the BEAR physics-based building simulation model based on the RC model with an nonlinear residual model. BEAR enables three user inputs: Building type, Weather type, and City. Users can either choose from a pre-defined list of buildings and climate types provided by Building Energy Codes Program [13] (See Appendix B) or define a customized BEAR environment by importing any self-defined EnergyPlus building models and weather files. BEAR also explicitly incorporates the nonlinear and stochastic heat transfer caused by building occupancy, making it flexible in considering various building usage and control scenarios.

We illustrate our simulator design using a two-zone, one-story building model as an example, as shown in Fig 1. ZONE 1 is entirely inside ZONE 2; thus, only ZONE 2 has external walls that connect to the outside air. The heat transferred through the model is considered from the temperature difference between neighboring zones, occupants' activity (Q^a), global horizontal irradiance (Q^s), and HVAC systems (Q^h). Here we show the modeling process for ZONE 1 and ZONE 2 using the following differential equations based on the RC model [7]:

$$C_1 \frac{dT_1}{dt} = \frac{T_2 - T_1}{R_{1,2}} + \frac{T_G - T_1}{R_{1,G}} + Q_1^h + Q_1^a + Q_1^s; \tag{1a}$$

$$C_2 \frac{dT_2}{dt} = \frac{T_E - T_2}{R_{E,2}} + \frac{T_1 - T_2}{R_{1,2}} + \frac{T_G - T_2}{R_{2,G}} + Q_2^h + Q_2^a + Q_2^s, \tag{1b}$$

where T_i is Zone i's temperature, T_G and T_E denote the ground and outdoor environment temperature. C_i is the thermal capacitance, $R_{i,j}$ is the thermal resistance between Zone i and Zone j and is symmetric, i.e., $R_{i,j} = R_{j,i}$. Q_i^h is the controlled heating supplied to

each zone; Q_i^a is the heat gained from indoor people activities; Q_i^s is the solar heat gained from windows for Zone i.

For a general building model with M indoor zones $i = 1, 2, ..., M$, the zone thermal dynamics are as follows:

$$C_i \frac{dT_i}{dt} = \sum_{j \in \mathcal{N}(i)} \frac{T_j - T_i}{R_{i,j}} + Q_i^h + Q_i^a + Q_i^s, \tag{2}$$

where $\mathcal{N}(i)$ are the neighboring zones of zone i. We encode each zone's connectivity to ensure only ground floor zones are connected to Zone G (ground), and peripheral zones are connected to Zone E (outdoor environment).

Heat Transfer Modeling: The following equations are descriptions of heat gained from different sources:

$$Q_i^h = w_i Q_i^z; \quad Q_i^a = n_i Q_p; \quad Q_i^s = \alpha_{SHGC} A_i^{win} Q_{ghi}, \tag{3}$$

where w_i represents the zonal HVAC efficiency coefficient, and Q_i^z denotes the heat gained from HVAC to compute the controlled zone heating. For modeling the heat gain from human activities Q_i^a, n_i denotes the number of people in each zone, and Q_p is the sensible heat gained from activities by one person. To model the solar heat, α_{SHGC} refers to the solar heat gain coefficient for windows, A_i^{win} is the zonal window area, and Q_{ghi} is the heat absorbed from global horizontal irradiance. Both Q_i^a and Q_i^s are time-varying uncontrolled heat generated from the environment, while Q_i^h is the controllable heat that could be taken as inputs from the simulator.

Here we calculate sensible heat per person Q_p using a polynomial function detailed in the EnergyPlus documentation [12]:

$$Q_p = c_1 + c_2 m + c_3 m^2 + c_4 \bar{T} - c_5 \bar{T} m + c_6 \bar{T} m^2 - c_7 \bar{T}^2 + c_8 \bar{T}^2 m - c_9 \bar{T}^2 m^2, \tag{4}$$

where m is the metabolic rate, $\bar{T} = \frac{1}{M}(T_1 + T_2 + ... + T_M)$ is the average zone temperature, and $c_1, ..., c_9$ are constants generated by fitting sensible heat data under varying conditions.

State Evolution: to simulate the building with designed control inputs, we re-organize the system dynamics model in (1)-(4) to the state-space form:

$$\dot{x}(t) = Ax(t) + Bu(t) + Df(t, x(t), r(t)), \tag{5}$$

where the state variable represents the collection of zone temperature, $x(t) = [T_1 \ T_2 \ \cdots \ T_M]^\top$, the control variable $u(t) = [T_G \ T_E \ Q_1^z \ Q_2^z \ \cdots \ Q_M^z \ Q_{ghi}]^\top$, and the nonlinear function for sensible heat calculation $f(t, x(t), r(t)) = c_1 + c_2 r(t) + c_3 r(t)^2 -$

$c_5 \bar{T} r(t) + c_6 \bar{T} r(t)^2 - c_7 \bar{T}^2 + c_8 \bar{T}^2 r(t) - c_9 \bar{T}^2 r(t)^2$, where $r(t)$ is the current metabolic rate m. The state matrix is as follows

$$A = \begin{bmatrix} \sum_{j \in \mathcal{N}(1)} \frac{-1}{C_1 R_{1,j}} + \frac{c_4 n_1}{MC_1} & \frac{1}{C_1 R_{1,2}} + \frac{c_4 n_1}{MC_1} & \cdots & \frac{1}{C_1 R_{1,M}} + \frac{c_4 n_1}{MC_1} \\ \frac{1}{C_2 R_{2,1}} + \frac{c_4 n_2}{MC_2} & \sum_{j \in \mathcal{N}(2)} \frac{-1}{C_2 R_{2,j}} + \frac{c_4 n_2}{MC_2} & \cdots & \frac{1}{C_2 R_{2,M}} + \frac{c_4 n_2}{MC_2} \\ \vdots & & \ddots & \vdots \\ \frac{1}{C_M R_{M,1}} + \frac{c_4 n_M}{MC_M} & \cdots & \cdots & \sum_{j \in \mathcal{N}(M)} \frac{-1}{C_M R_{M,j}} + \frac{c_4 n_M}{MC_M} \end{bmatrix},$$

the input matrix is

$$B = \begin{bmatrix} \frac{1}{C_1 R_{1,G}} & \frac{1}{C_1 R_{1,E}} & \frac{w_1}{C_1} & 0 & \cdots & 0 & \frac{\alpha_{SHGC} A_1^{win}}{C_1} \\ \frac{1}{C_2 R_{2,G}} & \frac{1}{C_2 R_{2,E}} & 0 & \frac{w_2}{C_2} & \cdots & 0 & \frac{\alpha_{SHGC} A_2^{win}}{C_2} \\ \vdots & \vdots & \vdots & & \ddots & \vdots & \vdots \\ \frac{1}{C_M R_{M,G}} & \frac{1}{C_M R_{M,E}} & 0 & 0 & \cdots & \frac{w_M}{C_M} & \frac{\alpha_{SHGC} A_M^{win}}{C_M} \end{bmatrix},$$

and matrix D in Eq (5) is calculated as $D = [\frac{n_1}{C_1} \quad \frac{n_2}{C_2} \quad \cdots \quad \frac{n_M}{C_M}]^\top$. We convert the continuous-time system model into a discrete-time representation,

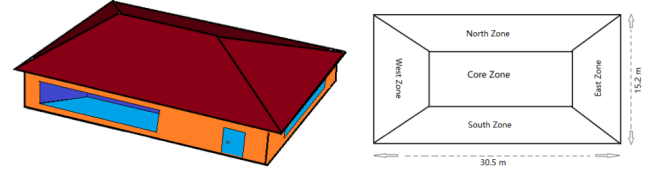$$x[k+1] = A_d x[k] + B_d u[k] + D_d F[k, x[k], r[k]], \quad (6)$$

where the term $A_d = e^{A \Delta T}$, ΔT is the sample time resolution. $B_d = A^{-1}(A_d - I)B$, $D_d = A^{-1}(A_d - I)D$ and $F[k, x[k], r[k]] = \int_{k\Delta T}^{(k+1)\Delta T} f(\tau, x(\tau), r(\tau)) d\tau$.

BEAR enables an automated pipeline to process building geometry, weather, and occupancy information to the discrete-time state-space models. Building parameters are obtained through the user-input building information. For example, R is determined by the wall material and volume, C is estimated by the volume of each zone, Q is computed by a combination of occupancy, GHI, and VAV information. Compared to the actual building model, our model in Eq (6) makes several simplifications regarding the zone shape, the window/door open schedules, the heat transfer function of HVAC, and the shadowing function. Detailed model assumptions are listed in the linked code repository. Nevertheless, extending the pre-defined dynamics to building use cases with user-defined zone shape, schedules, and shadowing functions is adaptable with our open-source building simulation engine.

Data-driven model: In some practical scenarios, the building parameters are not known exactly a priori, while only historical power consumption and temperature measurements are available. To address such gap between the simulation parameters and the actual building dynamics, we also incorporate a data-driven module in BEAR. We use Linear Regression to fit a data-driven building model $\hat{Y} = WX + b$ with coefficients $\{W, b\}$ that minimize the residual sum of squares between the ground truth and the predicted building states. Users could train with state and action data collected from a particular building of interests with minimal efforts and use the data-driven building model for controller/RL algorithm design. Besides the default linear data-driven models, users can also define other types of data-driven building models in BEAR such as neural networks and run gradient descent to update the model estimates.

BEAR's data-driven module takes the current step state-action pair $x[k], u[k], r[k]$ as input and predicts the next time-step state $x[k+1]$. To address the non-linear part of our model, we include $\bar{T}^2$ from the nonlinear function F as one of the input features, which could be calculated using $x[k]$. Thus, we collect $X = [x[k], u[k], \bar{T}^2]$ with target $Y = x[k+1]$.

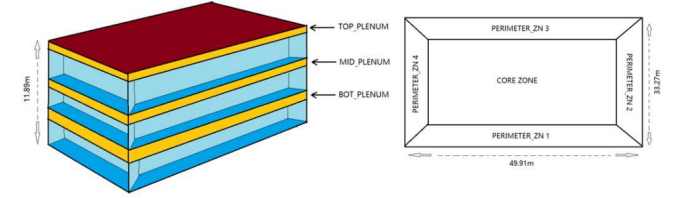A. Rectangular single-story Building

B. Office-Medium at Tucson

Figure 2: Simulated building examples.

3 RL ENVIRONMENT DESIGN

BEAR enjoys flexibility and high fidelity provided by a variety of user-defined variables as inputs (see details in Appendix C) and provides an OpenAI Gym interface. Users can perform simulations in the customized environment with any classic model-based control or learning-based controllers. A sample usage of BEAR package in Python is also illustrated in Appendix D.

State Space: The state s_k is the RL agent's observation from a building environment at timestep k. It is different from the state space model (6) by including both $x[k]$ and the uncontrollable inputs observed from the environment. The state space is bounded by user-defined minimum and maximum values. The state s_k is constructed as:

$$s_k = [T_1[k], T_2[k], ..., T_M[k], Q_p[k], T_G[k], T_E[k], Q_{ghi}[k]], \quad (7)$$

Action space: The action a_k is generated by the controller given state s_k. The action a_k is a set of controllable actions constructed with the energy supply of the HVAC system, as shown below:

$$a_k = [Q_1^z[k], Q_2^z[k], ..., Q_M^z[k]], \quad (8)$$

The whole action space is constrained by the maximum HVAC power consumption and normalized within the box of [-1, 1] in the user interface. All actions are rescaled to their original values inside the BEAR simulation. Once an action is selected, the $env.step()$ function provided by OpenAI Gym will take both s_k and a_k as input, and simulate the next state s_{k+1} using BEAR.

Reward: A main objective of building control is to reduce the energy consumption while keeping the temperature within given comfort range. Our platform allows users to customize reward function using environment states s_k, actions a_k, target values obj_k and a weight term β. We denote such reward function as $Reward = R(s_k, a_k, obj_k, \beta)$. BEAR users can customize reward by changing the weighting term, with small β leading to low energy consumption and large β leading to small temperature range deviation. One default reward function is the $L2$ reward, defined as

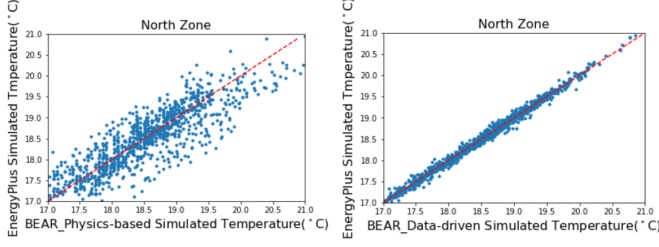$$R_k = -(1 - \beta)||a_k||_2 - \beta||s_k^{obj} - s_k||_2$$

Figure 3: Comparison of zonal temperature using BEAR simulator and EnergyPlus on the same test building located in Chicago, IL.

where $s_k^{obj} = [T_1^{obj}[k], ..., T_M^{obj}[k]]$ are the target temperature from user input.

4 EXPERIMENTS

In this section, we demonstrate the usage of BEAR with two building examples. We compare our simulator with EnergyPlus on a rectangular single-story building to validate the fidelity of our simulations. We also compare the performance of different control strategies, including rule-based controller, MPC (which knows the exact building dynamics model), and two RL controllers SAC [5], PPO [16] (which do not assume any model knowledge) on a simulated medium office building. Since our goal is to show the compatibility of our platform with multiple controllers, we directly set up all RL algorithms using Stable-Baselines 3 [14]. Detailed variable settings and building type examples can be found in our anonymous code repository.

Single-Story Building: To illustrate the fidelity of our simulator, we first set up a test example building model described in the EnergyPlus documentation and benchmark it against BEAR's simulations. See Fig 2 (A) for the building illustration. Specifically, we set up the physics-based model with building parameters estimated through information provided in the EnergyPlus. We also fit a data-driven model with data simulated by the EnergyPlus. A comparison of the zonal temperature simulated in BEAR and EnergyPlus without any HVAC control is shown in Fig 3. We could see both physics-based and data-driven simulation engine demonstrate good fidelity compared to EnergyPlus. Then, we set the indoor temperature at 22°C with a daily operating schedule of 8 a.m. to 3 p.m. In Fig 4, we compare the south zone temperature between our simulator and EnergyPlus using the same control input actions. We also compare the power demand by controlling each zone in the building strictly at the same target temperature. As is shown in Fig 5, under the same operational goal, we validate that not only the daily temperature in both simulations exhibit the same patterns (left), but the energy consumption profile of BEAR also closely tracks EnergyPlus's simulated trajectory (right).

Medium Office Building: We also test a medium office building provided by the reference commercial buildings list of the U.S. Department of Energy [4]. The building has three stories; each is divided into four perimeter zones, one core zone, and one plenum zone, see Fig 2 (B). The HVAC system is operated at all perimeter zones, and we set the indoor temperature of the perimeter zones at 22°C. We set the location of the building in Tucson, Arizona, and
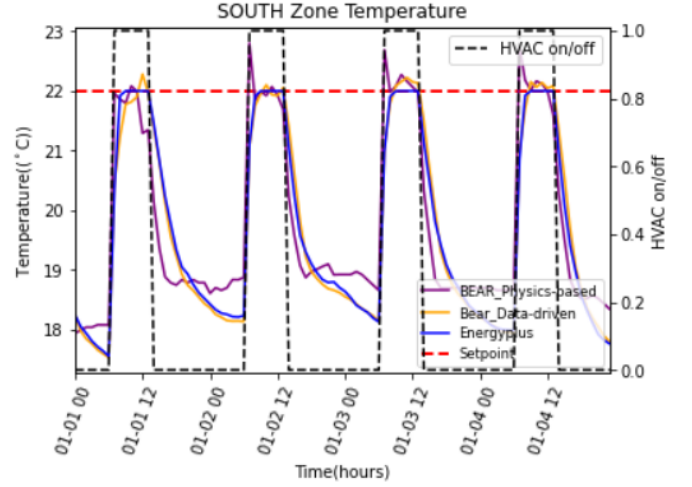

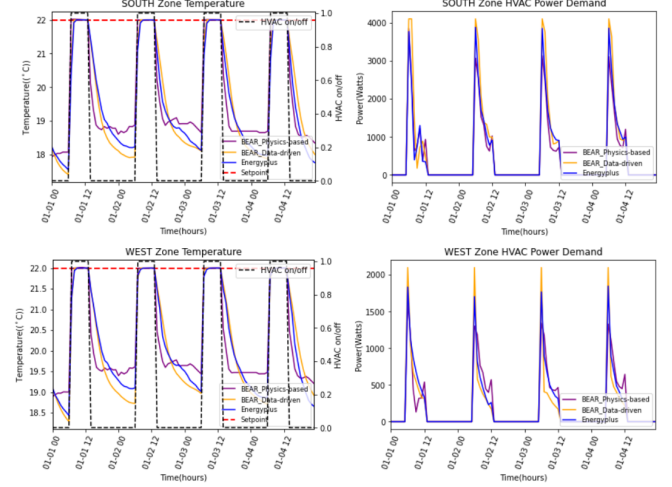**Figure 4: Temperature profile comparison with same control.**

Figure 5: Comparison of energy consumption.

Controller	Average Temperature Variation(°C)	Average Daily Energy Consumption (J)	Computation Time (s)
Rule-Based Controller	2.537	1.490E6	0.698
MPC ($\beta = 0.8$)	2.701E-11	6.25E5	33.572
PPO ($\beta = 0.8$)	0.969	6.504E5	1.309
SAC ($\beta = 0.8$)	0.795	6.188E5	1.348
MPC ($\beta = 0.45$)	0.383	5.87E5	33.633
PPO ($\beta = 0.45$)	2.645	4.680E5	1.339
SAC ($\beta = 0.45$)	1.360	5.691E5	1.235

Table 2: Performance on medium office.

use the weather file of 2003 from January to April for simulation. The control agents we tested include a rule-based controller, a MPC, and two RL controllers, namely PPO and SAC. The rule-based controller performs heating when the indoor temperature is below the target setpoint and performs cooling when the temperature is above the setpoint. The MPC, PPO, and SAC controllers are tested with two L_2 reward functions with $\beta = 0.8$ and $\beta = 0.45$ respectively. The learning curves of the RL controllers are shown in

Appendix A in Fig 6. We could see that all the RL training rewards are converging, and algorithms with the same reward function converge to a similar objective value. The performance of tested controllers is shown in Table 2. With the implementation of RL algorithms, we can observe that the energy consumption decreases significantly compared to the simple rule-based controller. It also reduces the average temperature variation. Compared to the MPC method, which has complete model knowledge, both model-free RL algorithms obtain a similar or lower energy consumption, with higher temperature violation. The computation time using PPO and SAC is greatly reduced compared to MPC, as the latter needs to solve an optimization problem at each step to obtain action.

5 CONCLUSION

This paper presents BEAR, an open-source physics-principled building control and RL platform compatible with OpenAI Gym. Unlike many existing platforms that use co-simulation with outsourcing building simulators, our platform is self-contained and thus provides simplicity for learning algorithm development and customized tasks. We illustrate the flexibility and efficiency of BEAR and various usage under both physics-based and data-driven settings. We plan to fill the gap between our building model and the real-world buildings by addressing factors such as shadow and light. We also plan to extend BEAR by supporting multi-agent RL training and heterogeneous reward design.

REFERENCES

[1] Javier Arroyo, Carlo Manna, Fred Spiessens, and Lieve Helsen. 2021. An OpenAI-Gym Environment for the Building Optimization Testing (BOPTEST) Framework. In *Proceedings of the 17th IBPSA Conference.*
[2] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym. *arXiv preprint arXiv:1606.01540* (2016).
[3] Drury B Crawley, Linda K Lawrie, Curtis O Pedersen, and Frederick C Winkelmann. 2000. Energy plus: energy simulation program. *ASHRAE journal* 42, 4 (2000), 49–56.
[4] Michael Deru, Kristin Field, Daniel Studer, Kyle Benne, Brent Griffith, Paul Torcellini, Bing Liu, Mark Halverson, Dave Winiarski, Michael Rosenberg, et al. 2011. US Department of Energy commercial reference building models of the national building stock. (2011).
[5] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning.* PMLR, 1861–1870.
[6] Javier Jiménez-Raboso, Alejandro Campoy-Nieves, Antonio Manjavacas-Lucas, Juan Gómez-Romero, and Miguel Molina-Solana. 2021. Sinergym: a building simulation and control framework for training reinforcement learning agents. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation.* 319–323.
[7] Yudong Ma, Anthony Kelman, Allan Daly, and Francesco Borrelli. 2012. Predictive control for energy efficient buildings with thermal storage: Modeling, stimulation, and experiments. *IEEE control systems magazine* 32, 1 (2012), 44–64.
[8] Karl Mason and Santiago Grijalva. 2019. A review of reinforcement learning for autonomous building energy management. *Computers & Electrical Engineering* 78 (2019), 300–312.
[9] Sven Erik Mattsson and Hilding Elmqvist. 1997. Modelica-An international effort to design the next generation modeling language. *IFAC Proceedings Volumes* 30, 4 (1997), 151–155.
[10] Takao Moriyama, Giovanni De Magistris, Michiaki Tatsubori, Tu-Hoa Pham, Asim Munawar, and Ryuki Tachibana. 2018. Reinforcement learning testbed for power-consumption optimization. In *Asian simulation conference.* Springer.
[11] Stephen Nalley and Angelina LaRose. 2021. Annual energy outlook 2021. *United States Energy Information Administration: Washington DC* (2021).
[12] U.S. Department of Energy. 2022. Engineering Reference. https://energyplus.net/assets/nrel_custom/pdfs/pdfs_v22.1.0/EngineeringReference.pdf.
[13] The Building Energy Codes Program. [n.d.]. Prototype building models. https://www.energycodes.gov/prototype-building-models.
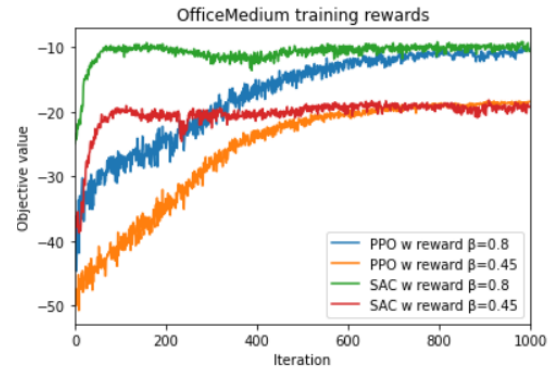

Figure 6: Reward during Training for RL policies.

[14] Antonin Raffin, Ashley Hill, Maximilian Ernestus, Adam Gleave, Anssi Kanervisto, and Noah Dormann. 2022. Stable-Baselines3. https://github.com/DLR-RM/stable-baselines3/blob/master/docs/index.rst.
[15] Paul Scharnhorst, Baptiste Schubnel, Carlos Fernández Bandera, Jaume Salom, Paolo Taddeo, Max Boegli, Tomasz Gorecki, Yves Stauffer, Antonis Peppas, and Chrysa Politi. 2021. Energym: A building model library for controller benchmarking. *Applied Sciences* 11, 8 (2021), 3518.
[16] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).
[17] Rui Tang and Shengwei Wang. 2019. Model predictive control for thermal energy storage and thermal comfort optimization of building demand response in smart grids. *Applied Energy* 242 (2019), 873–882.
[18] José R Vázquez-Canteli, Sourav Dey, Gregor Henze, and Zoltán Nagy. 2020. CityLearn: Standardizing research in multi-agent reinforcement learning for demand response and urban energy management. *preprint arXiv:2012.10504* (2020).
[19] Michael Wetter, Philip Haves, and Brian Coffey. 2008. *Building controls virtual test bed.* Technical Report. Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States).
[20] Zhiang Zhang and Khee Poh Lam. 2018. Practical implementation and evaluation of deep reinforcement learning control for a radiant heating system. In *Proceedings of the 5th Conference on Systems for Built Environments.* 148–157.

A RL TRAINING CURVE

Training curve for the PPO and SAC algorithms with different rewards in controlling the medium office building in figure 6 .

B LIST OF SIMULATING CONFIGURATIONS

The prototype buildings included in BEAR are derived from DOE's Commercial Reference Building Models. The models include 16 commercial building types in 19 locations. Users can download the models at https://www.energycodes.gov/prototype-building-models. Since BEAR is compatible with EnergyPlus, users could also create their own model in EnergyPlus editor and load the generated table file into BEAR.

- *Available building types*: ApartmentHighRise, ApartmentMidRise, Hospital, HotelLarge, HotelSmall, OfficeLarge, OfficeMedium, OfficeSmall, OutPatientHealthCare, RestaurantFastFood, RestaurantSitDown, RetailStandalone, RetailStripmall, SchoolPrimary, SchoolSecondar, Warehouse.
- *Available weather types*: Very Hot Humid, Hot Humid, Hot Dry, Warm Humid, Warm Dry, Warm Marine, Mixed Humid, Mixed Dry, Mixed Marine, Cool Humid, Cool Dry, Cool Marine, Cold Humid, Cold Dry, Very Cold, Subarctic/Arctic.

Input Variables	Type	Description	Default value	Source
Filename	Str	Filename of the selected building model	Required	Building List or User-defined
Weatherfile	Str	Filename of the selected weather	Required	Weather list
Location	List with length of 12	Ground temperatures of 12 month	Required	Ground Temperature Dictionary
U-Wall	List with length of 7	U-factors of Walls	Given by building model	Building list
Target	List with length of zone number	Target temperature of comfort	[22°C...22°C]	User-defined
Time Reso	Int	Length of one time-step	3600 second	User-defined
Reward-gamma	List with length of 2	Weight for comfort level and energy demand	[0.001,0.999]	User-defined
SHGC	Int	Solar Heat Gain Coefficient	0.252	User-defined
SHGC-weight	Int	Radiative/convective split for heat gain	0.1	User-defined
Ground-weight	Int	Lost of heat gain from ground	0.5	User-defined
Full-Occ	List with length of zone number	Number of people in each zone	[0...0]person	User-defined
Activity-sch	List with length of the simulation	The activity schedule of people	[120...120] W/person	User-defined
AC-map	List of Boolean with length of zone number	Map of HVAC in the building	[1...1]	User-defined
Max-power	Int	Maximum power of HVAC	8000 W	User-defined

Table 3: Variables table.

- *Available locations*: Albuquerque, Atlanta, Buffalo, Denver, Dubai, ElPaso, Fairbanks, GreatFalls, HoChiMinh, Honolulu, InternationalFalls, NewDelhi, NewYork, PortAngeles, Rochester, SanDiego, Seattle, Tampa, Tucson.

C VARIABLE TABLE

A large variety of variables could be defined by BEAR users. Three inputs (Filename, Weatherfile, Location) are required for setting up a basic building environment, while users could modify other variables for better simulation accuracy. Variable **U-WAll** contains the U values of each wall used in the building model, which could be changed if user would like to replace the material of the wall. Variable **Target** should be used for self-define control objectives. Variable **Time-Reso** should be modified if user would like to change the sample time of the simulation. Variable **Reward-gamma** can change the reward function coefficient. Variable **SHGC** is based on the materials of window. Both SHGC and Ground temperature could have partial impact on zone temperature depending on the building structure, thus **SHGC-weight** and **Ground-weight** are provided for user to tune. Variable **Full-Occ** is used to set up occupancy of each zone. Variable **Activity-sch** represents the metabolic heat of different activities. Variable **AC-map** and **Max-power** can be used to change the location and the maximum output of the HVAC system. To address the non-linear part of our model, we assume all people in the building perform similar activities, which guarantees a constant metabolic rate $r(t)$.

Table 3 summarizes the full list of variables currently implemented in BEAR.

D EXAMPLE USAGE

A simple usage example is shown in Fig 7. The objective of the code snippet is to simulate a 'SchoolPrimary' type building at Tucson with 'Hot Dry' weather using random selected actions. As a first step, an environment with the required building/weather/city is created with the building parameters generated. Once a building model is created, the detailed Zone information would be printed out. Then we start the simulation with the reset function to observe the initial state. At each step, the controller agent would observe the current state s_t, and generate a corresponding action a_t. The environment would then take the action and pass it into the building state-space model to simulate the new state s_{t+1} for the next timestep. A 24-hour simulation is performed in the for-loop in this

```python
data=pd.read_csv('DataDriven_training.csv')
trainstate=data['Zone Air Temperature [C](Hourly)']
trainaction = data['Zone Air System Sensible Heating Rate [W](Hourly)']

Parameter=ParameterGenerator('SchoolPrimary','Hot_Dry','Tucson')
#Create environment
env = BuildingEnvReal(Parameter)
numofhours=24
#Initialize
env.reset()
#Train with Data-driven module
env.train(np.array(trainstate).T, np.array(trainaction).T)
for i in range(numofhours):
    a = env.action_space.sample()#Randomly select an action
    obs, r, done, _ = env.step(a)#Return observation and reward
RandomController_state=env.statelist  #Collect the state list
RandomController_action=env.actionlist  #Collect the action list
env.close()
```

```
###############All Zones from Ground###########
CORNER_CLASS_1_POD_1_ZN_1_FLR_1 [Zone index]: 0
MULT_CLASS_1_POD_1_ZN_1_FLR_1 [Zone index]: 1
CORRIDOR_POD_1_ZN_1_FLR_1 [Zone index]: 2
CORNER_CLASS_2_POD_1_ZN_1_FLR_1 [Zone index]: 3
MULT_CLASS_2_POD_1_ZN_1_FLR_1 [Zone index]: 4
CORNER_CLASS_1_POD_2_ZN_1_FLR_1 [Zone index]: 5
MULT_CLASS_1_POD_2_ZN_1_FLR_1 [Zone index]: 6
CORRIDOR_POD_2_ZN_1_FLR_1 [Zone index]: 7
CORNER_CLASS_2_POD_2_ZN_1_FLR_1 [Zone index]: 8
MULT_CLASS_2_POD_2_ZN_1_FLR_1 [Zone index]: 9
CORNER_CLASS_1_POD_3_ZN_1_FLR_1 [Zone index]: 10
MULT_CLASS_1_POD_3_ZN_1_FLR_1 [Zone index]: 11
CORRIDOR_POD_3_ZN_1_FLR_1 [Zone index]: 12
CORNER_CLASS_2_POD_3_ZN_1_FLR_1 [Zone index]: 13
MULT_CLASS_2_POD_3_ZN_1_FLR_1 [Zone index]: 14
COMPUTER_CLASS_ZN_1_FLR_1 [Zone index]: 15
MAIN_CORRIDOR_ZN_1_FLR_1 [Zone index]: 16
LOBBY_ZN_1_FLR_1 [Zone index]: 17
MECH_ZN_1_FLR_1 [Zone index]: 18
BATH_ZN_1_FLR_1 [Zone index]: 19
OFFICES_ZN_1_FLR_1 [Zone index]: 20
GYM_ZN_1_FLR_1 [Zone index]: 21
KITCHEN_ZN_1_FLR_1 [Zone index]: 22
CAFETERIA_ZN_1_FLR_1 [Zone index]: 23
LIBRARY_MEDIA_CENTER_ZN_1_FLR_1 [Zone index]: 24
##################################################
```

Figure 7: Code for a sample usage of BEAR.

example case. Each loop would generate a random action, and send the action to the environment to observe the new state, reward, and termination.

A customized usage example for self-defined building is shown in Fig 8 to illustrate. In this code snippet, a new building not included in

```
chicago=[20.4, 20.4, 20.4, 20.4, 21.5, 22.7, 22.9, 23, 23, 21.9, 20.7, 20.5]
city='chicago'
filename='Exercise2A-mytestTable.html'
weatherfile='USA_IL_Chicago-OHare.Intl.AP.725300_TMY3.epw'
U_Wall=[2.811, 12.894, 0.408, 0.282, 1.533, 12.894, 1.493]
Parameter=ParameterGenerator(filename, weatherfile, city,
                             U_Wall=U_Wall, Ground_Tp=chicago, shgc=0.568)
env = BuildingEnvReal(Parameter)
numofhours=24*(4)
agent = MPCAgent(env, gamma=env.gamma,
                 safety_margin=0.96, planning_steps=10)
env.reset()
for i in range(numofhours):
    action, s = agent.predict(env)#MPC agent selects an action
    obs, r, done, _ = env.step(action)#Return observation and reward
MPCstate=env.statelist  #Collect the state list
MPCaction=env.actionlist#Collect the action list


###############All Zones from Ground############
SOUTH PERIMETER [Zone index]: 0
EAST PERIMETER [Zone index]: 1
NORTH PERIMETER [Zone index]: 2
WEST PERIMETER [Zone index]: 3
CORE [Zone index]: 4
PLENUM [Zone index]: 5
#################################################
```

Figure 8: Code for a customize usage of BEAR.

the provided building prototype list is shown. User can self-define a building through the EnergyPlus editor, and upload the EnergyPlus html file and the epw weather file into BEAR. In the example, the wall materials and ground temperatures are customized. SHGC value is also modified. During simulation, a MPC controller is used instead of a RL controller.